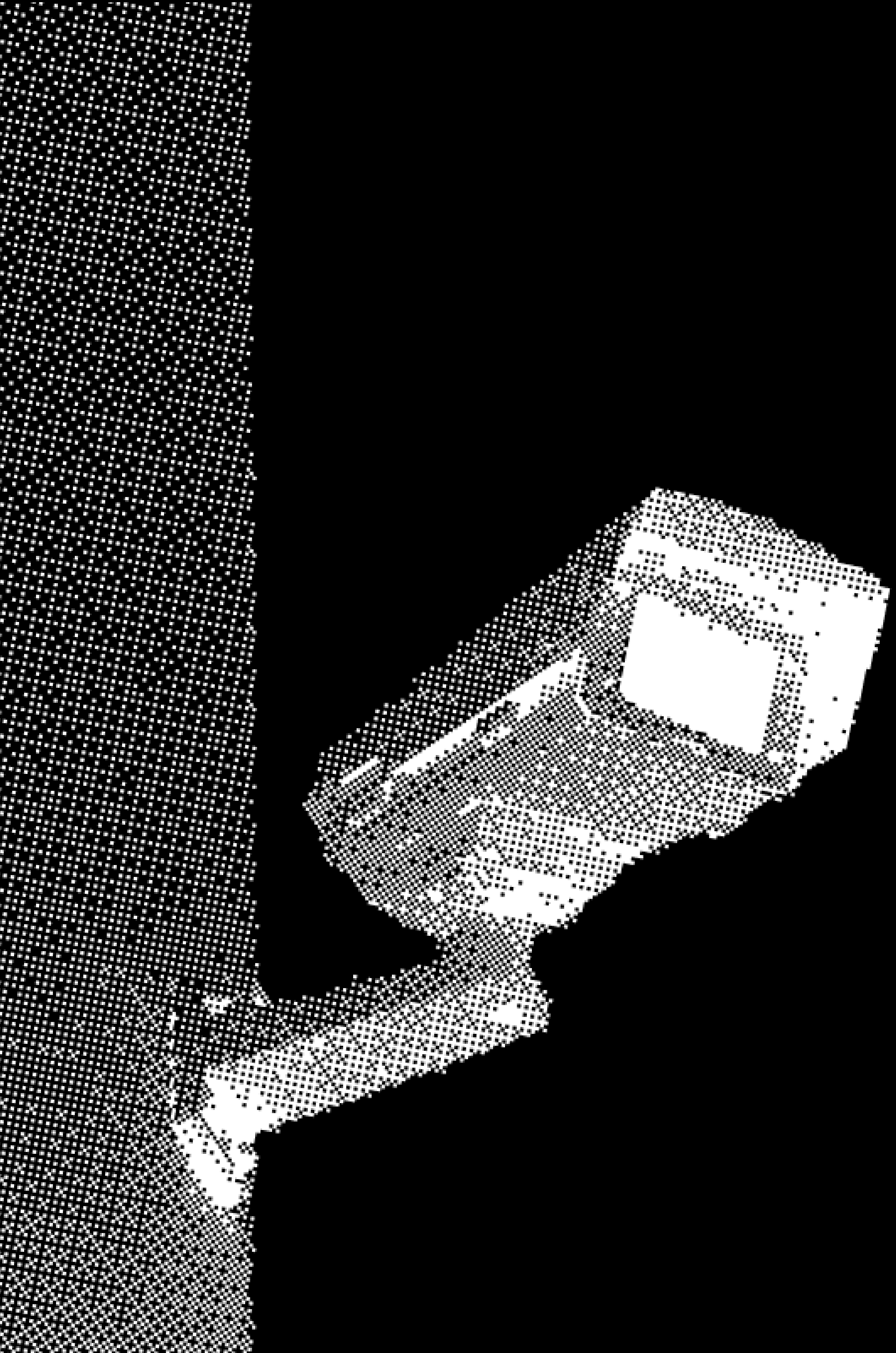
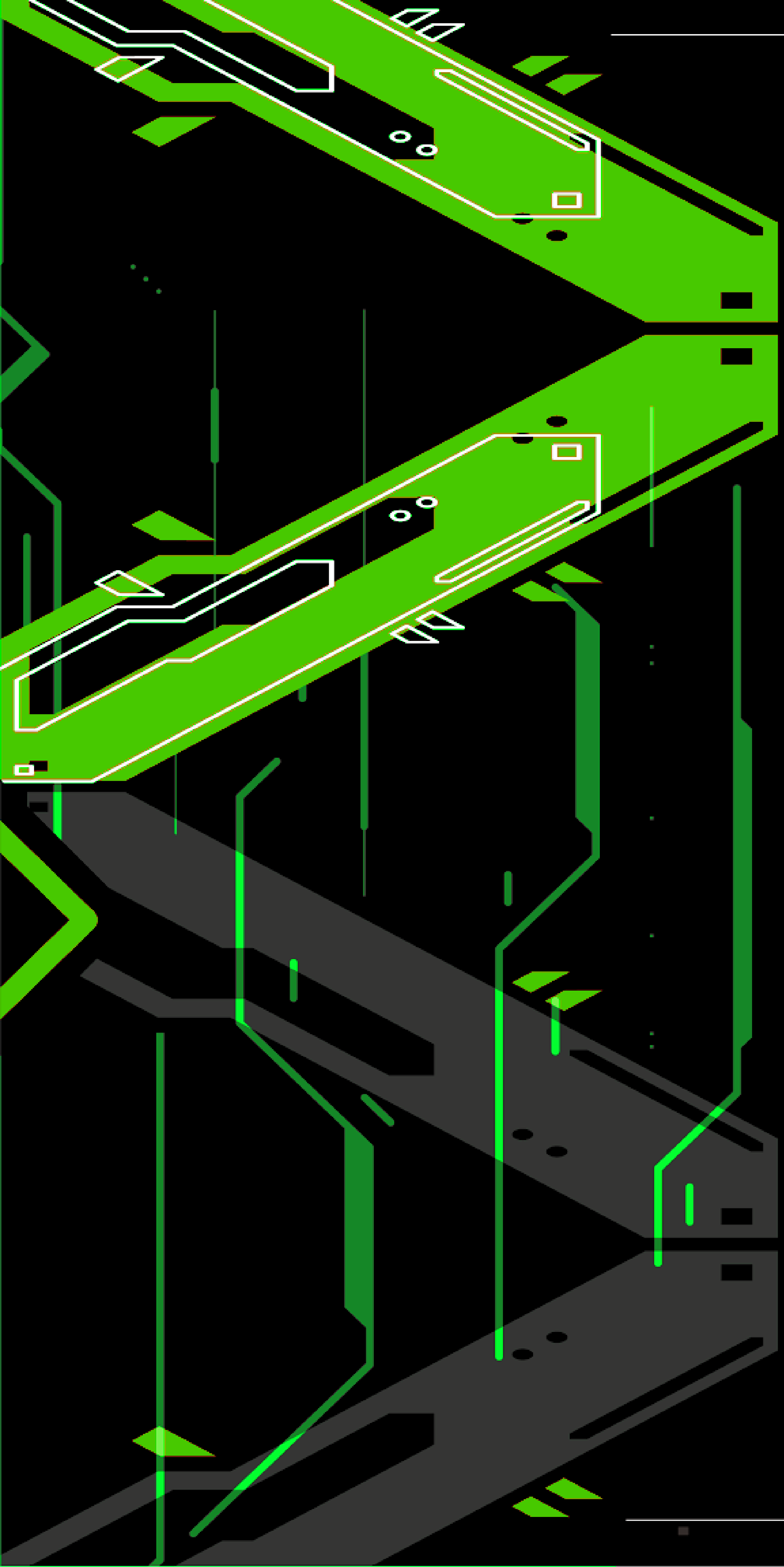


breaking web3 bridges

+ burning out any web2 hacker

author : six



talk agenda

the becoming of web3 hackers

bridge hacking techniques

- web2 attacks against "web3" systems
- ecdsa signature forgery
- social engineering

future price prediction

conclusion

```
$ id six
uid=1000(six) gid=1000(six)
groups=1001(independent hacker),
4(cctf founder),
7(qrucial dao co-founder)
24(polkadot head ambassador),
27(sudo)
```

workshop feedback

from infosec conferences

coin, token... bridge... a VM on the blockchain which you code runs...?

but this.. i can't understand...

there is a void here.

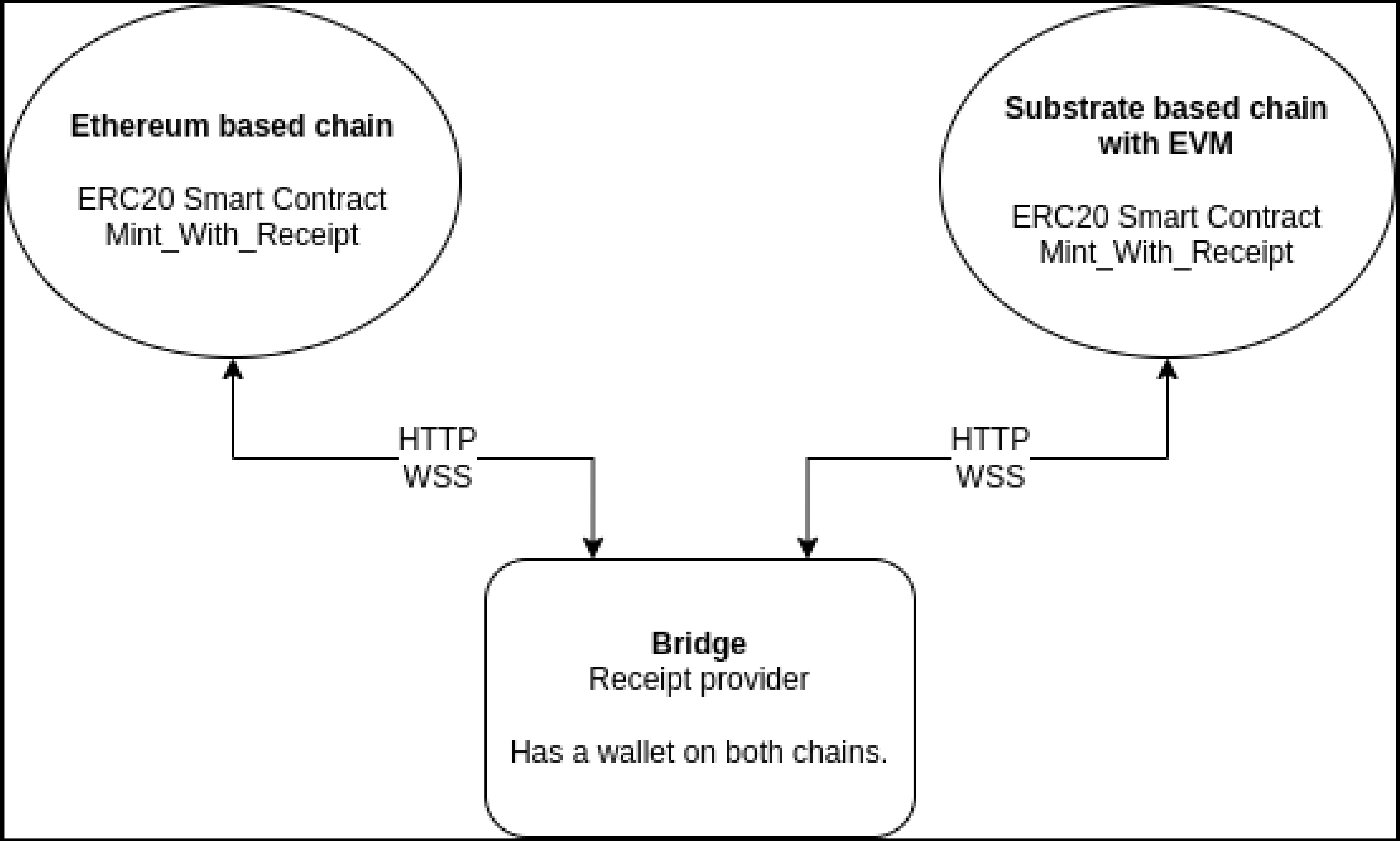
i couldn't imagine it...

i felt like a dinosaur.

inspiration

you are on the best scene

simple topology



Ethereum based chain

ERC20 Smart Contract
Mint_With_Receipt

Substrate based chain with EVM

ERC20 Smart Contract
Mint_With_Receipt

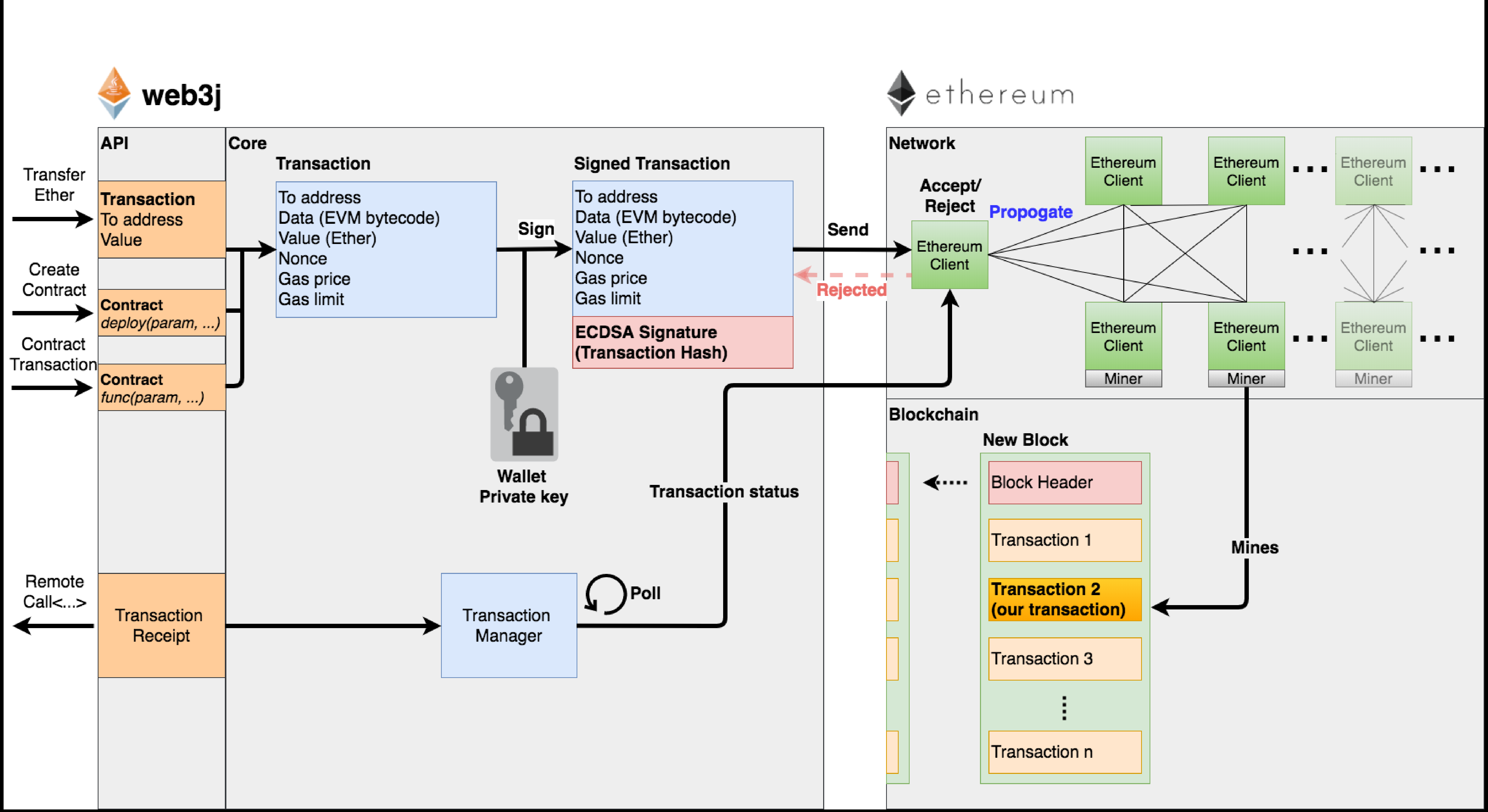
HTTP
WSS

HTTP
WSS

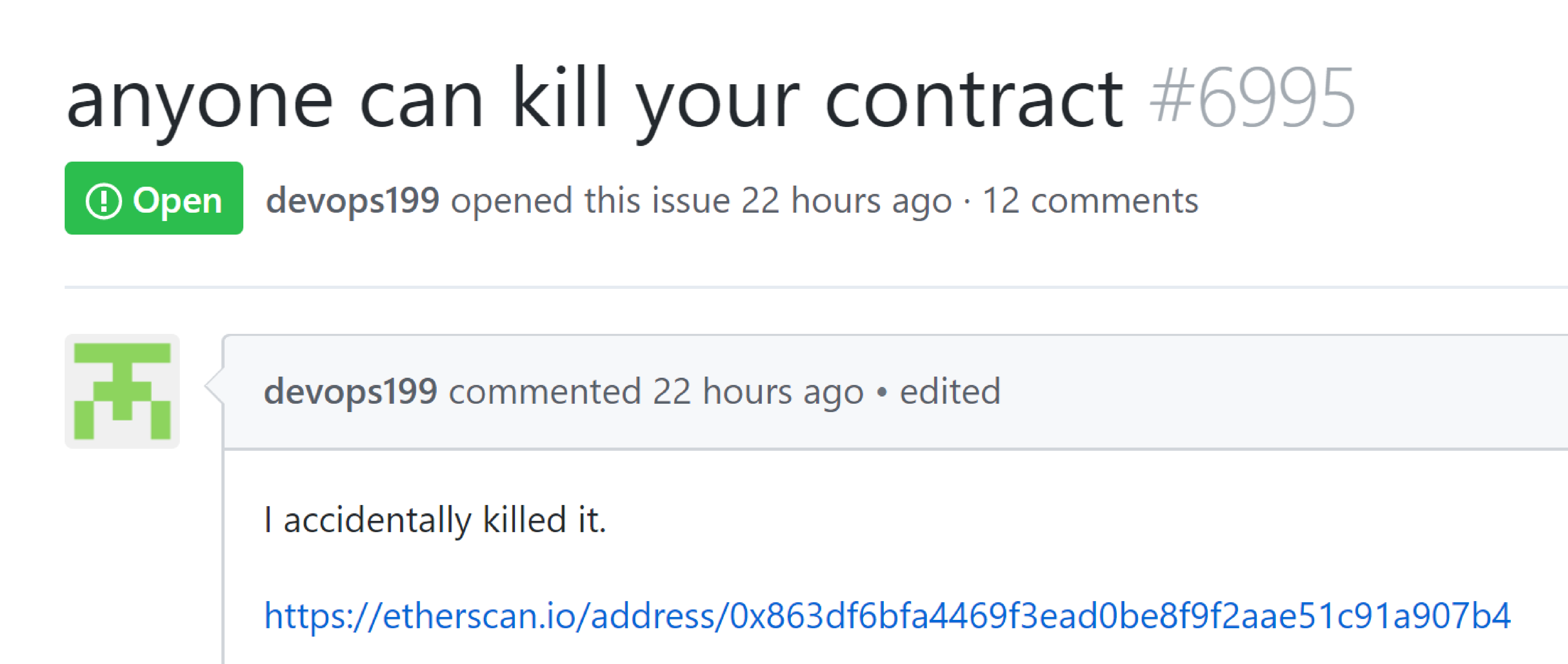
Bridge
Receipt provider

Has a wallet on both chains.

basic stuff: eth tx



warmup technique



anyone can kill your contract #6995

Open devops199 opened this issue 22 hours ago · 12 comments

devops199 commented 22 hours ago • edited

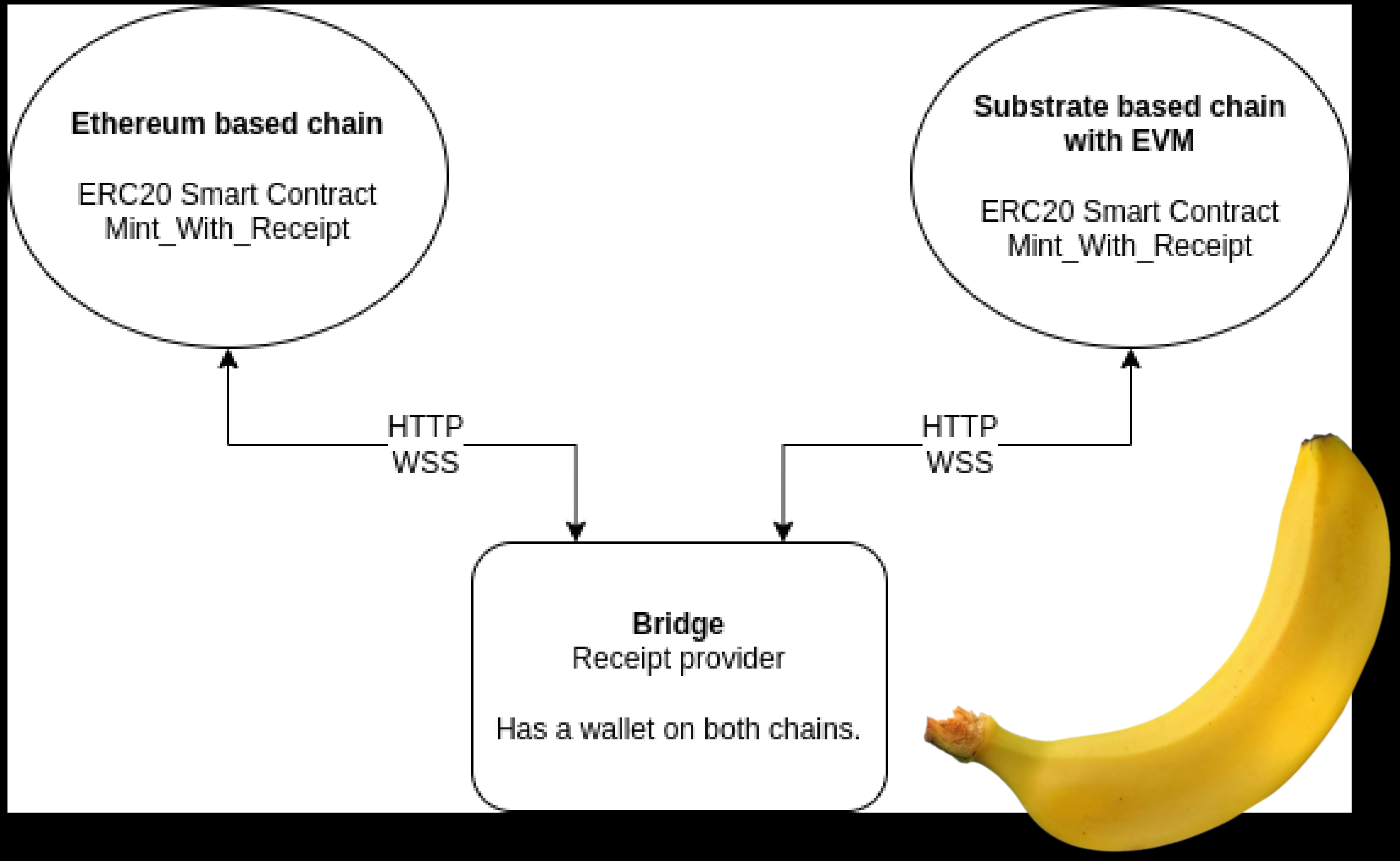
I accidentally killed it.

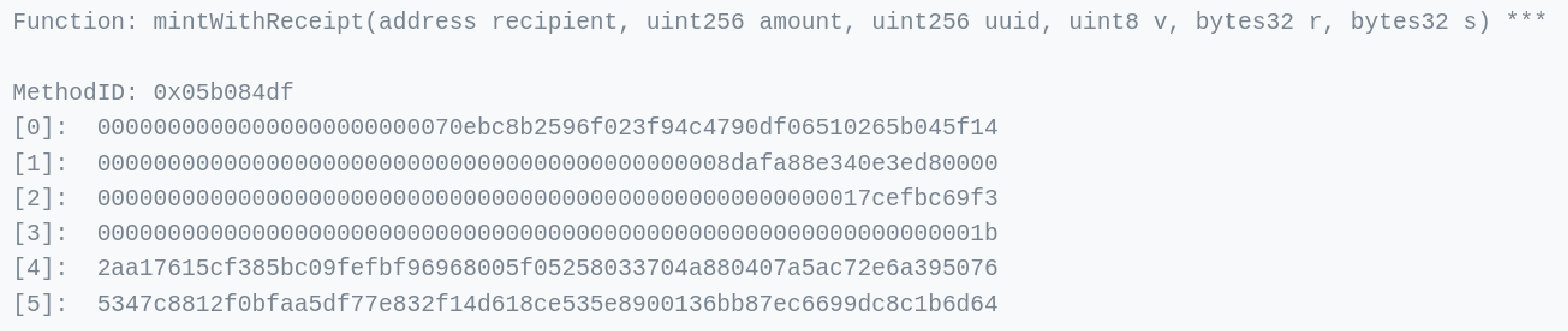
https://etherscan.io/address/0x863df6bfa4469f3ead0be8f9f2aae51c91a907b4

multisig issue: https://github.com/openethereum/parity-ethereum/issues/6995

txdata replay attack

"TRDR" (or tl;dr I didn't read the contract)





```
Function: mintWithReceipt(address recipient, uint256 amount, uint256 uuid, uint8 v, bytes32 r, bytes32 s) ***

MethodID: 0x05b084df
[0]:  00000000000000000000000070ebc8b2596f023f94c4790df06510265b045f14
[1]:  000000000000000000000000000000000000000000008dafa88e340e3ed80000
[2]:  00000000000000000000000000000000000000000000000000000017cefbc69f3
[3]:  000000000000000000000000000000000000000000000000000000000000001b
[4]:  2aa17615cf385bc09fefbf96968005f05258033704a880407a5ac72e6a395076
[5]:  5347c8812f0bfaa5df77e832f14d618ce535e8900136bb87ec6699dc8c1b6d64
```

wban hack txs:

https://polygonscan.com/tx/0xbcf3f1192d63a0d240995619b8896c406d1ba6fa7c2fc81503057d61c98bba41

https://bscscan.com/tx/0x60c3ae26d1a1d2b525a425aacdbde30bf7efdc09a125086cc7aab9b347daf684

ecdsa signature forgery

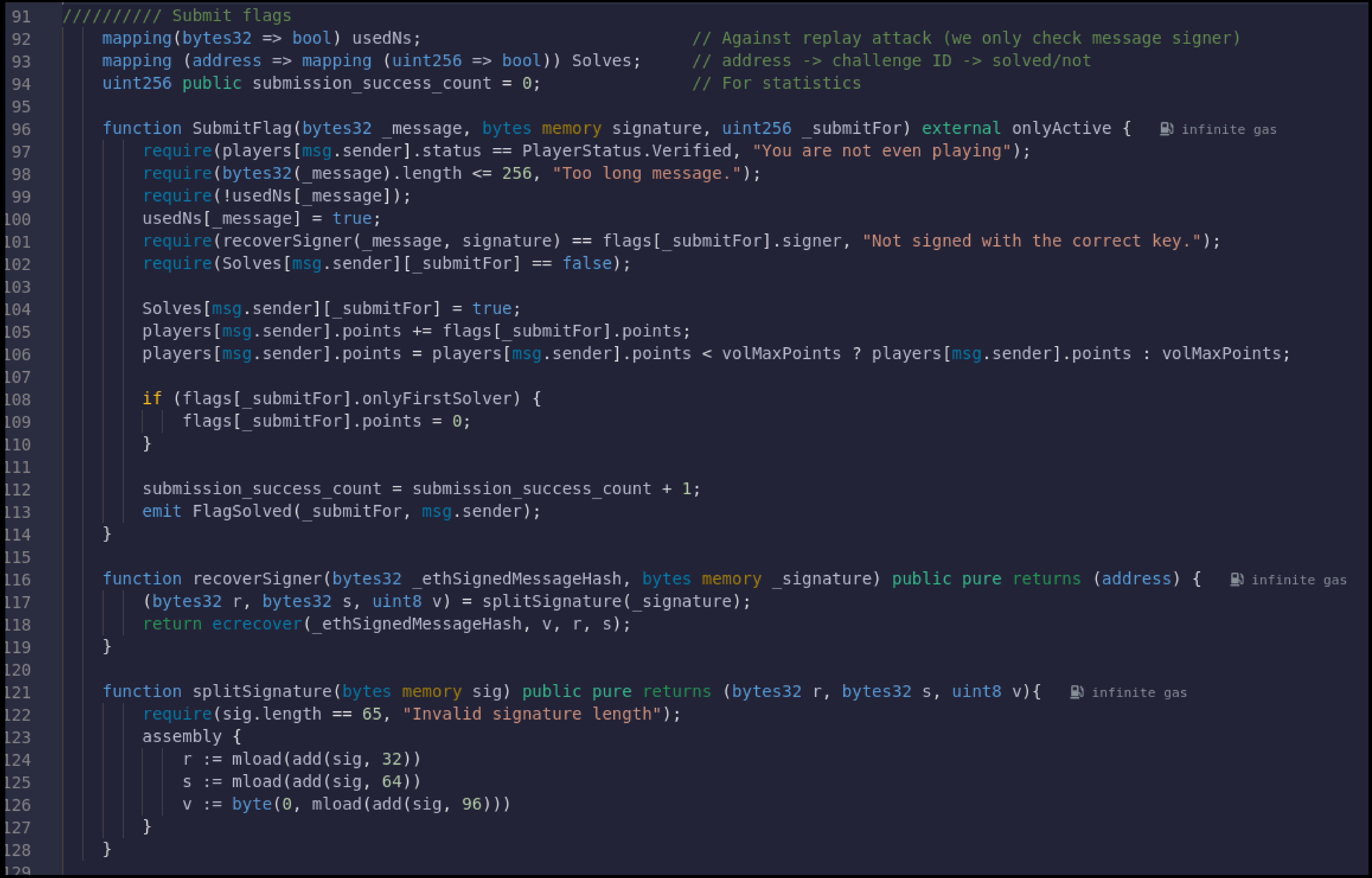
thanks for SI from CCTF

scope:
ElGamal-type digital signatures - ECDSA incl.

property:
signatures for any given pubkey, can be forged to unclean messages

vulnerability:
only the address is checked in smart contract.

poc:
https://git.hsbp.org/six/pwn_w3bridges



```
////////// Submit flags
    mapping(bytes32 => bool) usedNs;                          // Against replay attack (we only check message signer)
    mapping (address => mapping (uint256 => bool)) Solves;    // address -> challenge ID -> solved/not
    uint256 public submission_success_count = 0;              // For statistics

    function SubmitFlag(bytes32 _message, bytes memory signature, uint256 _submitFor) external onlyActive {
        require(players[msg.sender].status == PlayerStatus.Verified, "You are not even playing");
        require(bytes32(_message).length <= 256, "Too long message.");
        require(!usedNs[_message]);
        usedNs[_message] = true;
        require(recoverSigner(_message, signature) == flags[_submitFor].signer, "Not signed with the correct key.");
        require(Solves[msg.sender][_submitFor] == false);

        Solves[msg.sender][_submitFor] = true;
        players[msg.sender].points += flags[_submitFor].points;
        players[msg.sender].points = players[msg.sender].points < volMaxPoints ? players[msg.sender].points : volMaxPoints;

        if (flags[_submitFor].onlyFirstSolver) {
            flags[_submitFor].points = 0;
        }

        submission_success_count = submission_success_count + 1;
        emit FlagSolved(_submitFor, msg.sender);
    }

    function recoverSigner(bytes32 _ethSignedMessageHash, bytes memory _signature) public pure returns (address) {
        (bytes32 r, bytes32 s, uint8 v) = splitSignature(_signature);
        return ecrecover(_ethSignedMessageHash, v, r, s);
    }

    function splitSignature(bytes memory sig) public pure returns (bytes32 r, bytes32 s, uint8 v){
        require(sig.length == 65, "Invalid signature length");
        assembly {
            r := mload(add(sig, 32))
            s := mload(add(sig, 64))
            v := byte(0, mload(add(sig, 96)))
        }
    }
```

reminder: axie hack

problem:
blockchain games are not fully on blockchain nor decentralized

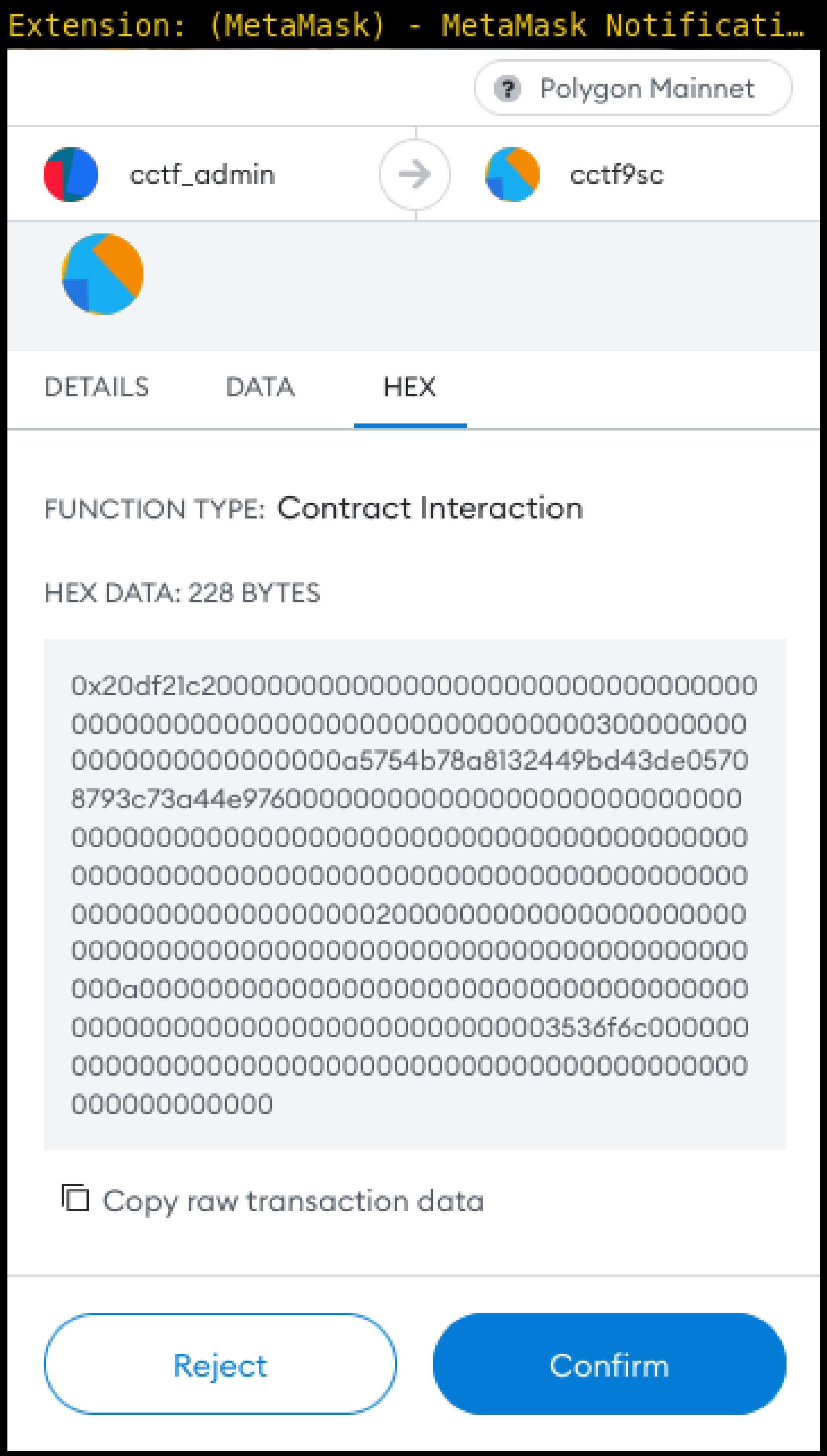
exploit:
web2 type of hacking leading to 51p attack

timeline:
6 days until axie realized they are being hacked

social engineer

<3

life of a standard metamask user

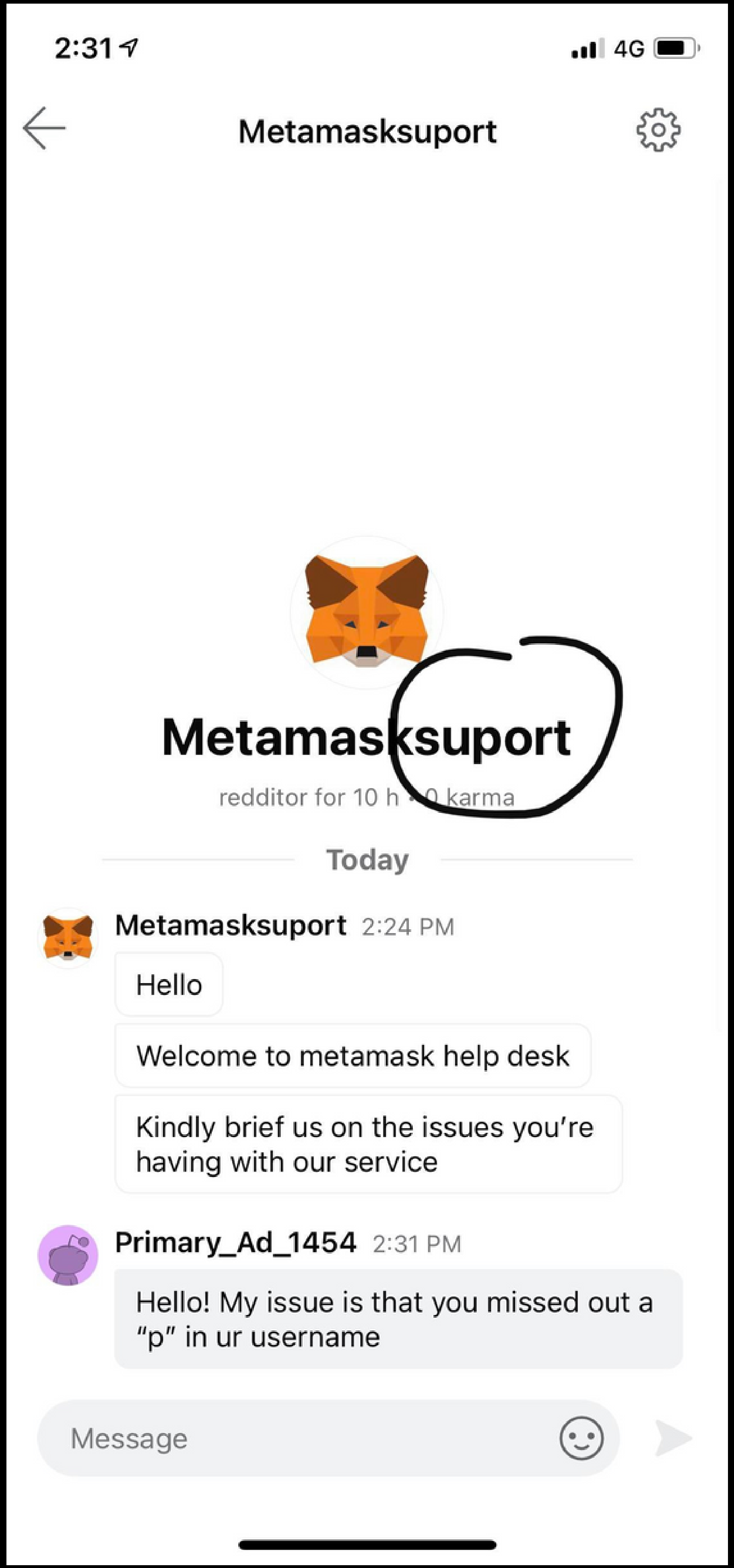


Extension: (MetaMask) - MetaMask Notificati…

FUNCTION TYPE: Contract Interaction

HEX DATA: 228 BYTES

```
0x20df21c200000000000000000000000000000000
0000000000000000000000000000000300000000
0000000000000000a5754b78a8132449bd43de0570
8793c73a44e9760000000000000000000000000000
0000000000000000000000000000000000000000
0000000000000000000000000000000000000000
0000000000000000000200000000000000000000
0000000000000000000000000000000000000000
000a000000000000000000000000000000000000
0000000000000000000000000003536f6c000000
0000000000000000000000000000000000000000
000000000000
```



Metamasksuport

Metamasksuport

redditor for 10 h • 0 karma

Today

Metamasksuport 2:24 PM

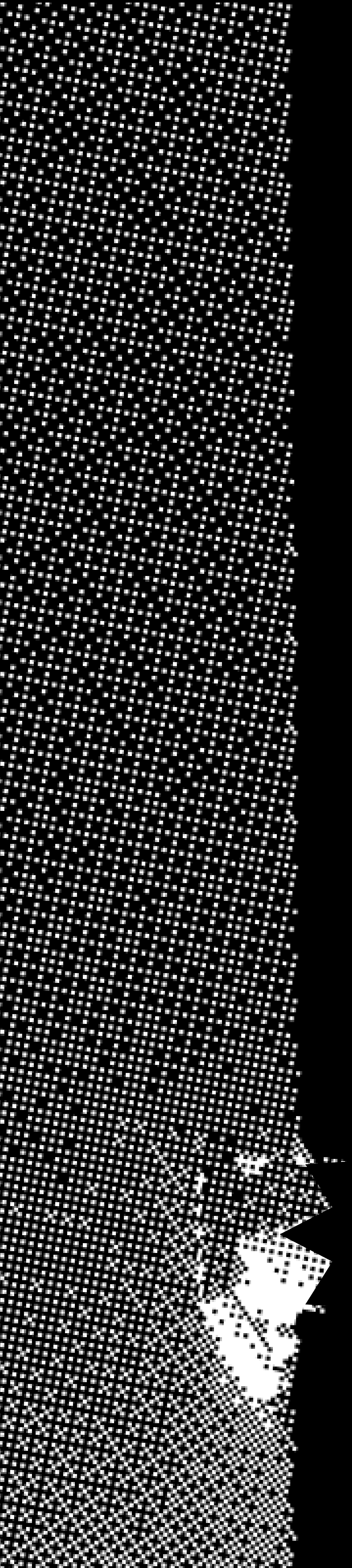
Hello

Welcome to metamask help desk

Kindly brief us on the issues you're having with our service

Primary_Ad_1454 2:31 PM

Hello! My issue is that you missed out a "p" in ur username



future price prediction

404

reminder: build for good, not for money

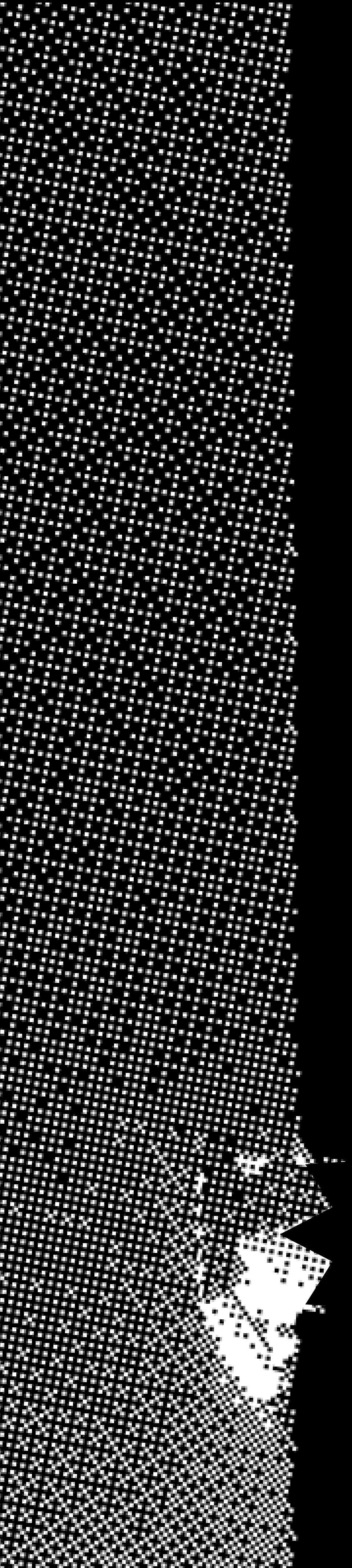
conclusion

keep in mind: we are in a highly experimental env

transparency + awareness ftw

gl & hf!

(and buy my sec audit services)



contact

sixthedave.me

matrix: @hexff:matrix.org

twitter: @SixTheDave

references

wrapped banano hack writeup:
https://medium.com/banano/wrapped-banano-wban-bridges-rekt-epilogue-85e4a31c16e2

elgamal type digital signatures:
https://coders-errand.com/malleability-ecdsa-signatures/
https://cryptoctf.org/2022/09/11/writeup-of-flag-submission-forgery-by-si/
https://github.com/Sunzehan/Project-forge-a-signature-to-pretend-that-you-are-Satoshi
https://gist.github.com/chjj/4fe8f5b2b489e89e6ed4
https://git.hsbp.org/six/eth_keygen

cctf challenges:
https://cryptoctf.org/